Message Integrity
and
Message Authentication

Objectives

- To define message integrity
- To define message authentication
- To distinguish between an MDC and a MAC
- To discuss some common MACs

MESSAGE INTEGRITY

- The cryptography systems that we have studied so far provide secrecy, or confidentiality, but not integrity.
- However, there are occasions where we may not even need secrecy but instead must have integrity.

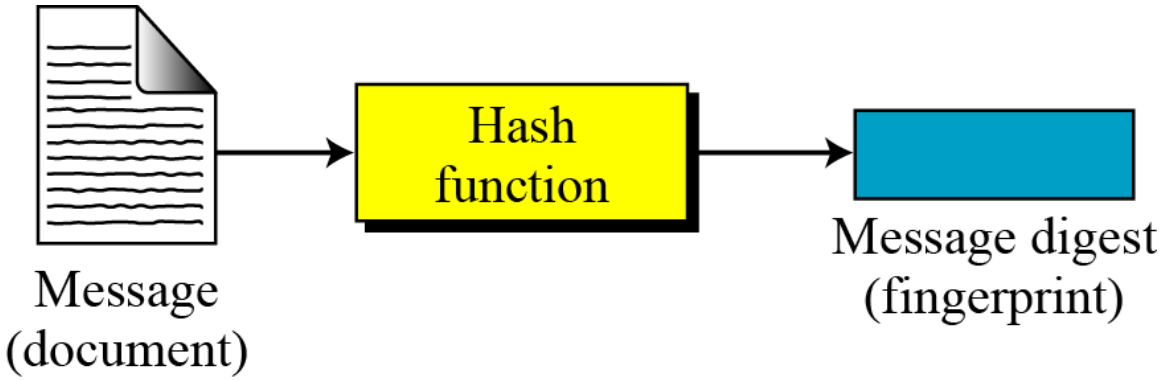
Document and Fingerprint

- One way to preserve the integrity of a document is through the use of a fingerprint.
- If Alice needs to be sure that the contents of her document will not be changed, she can put her fingerprint at the bottom of the document.

Message and Message Digest

The electronic equivalent of the document and fingerprint pair is the message and digest pair.

Message and digest

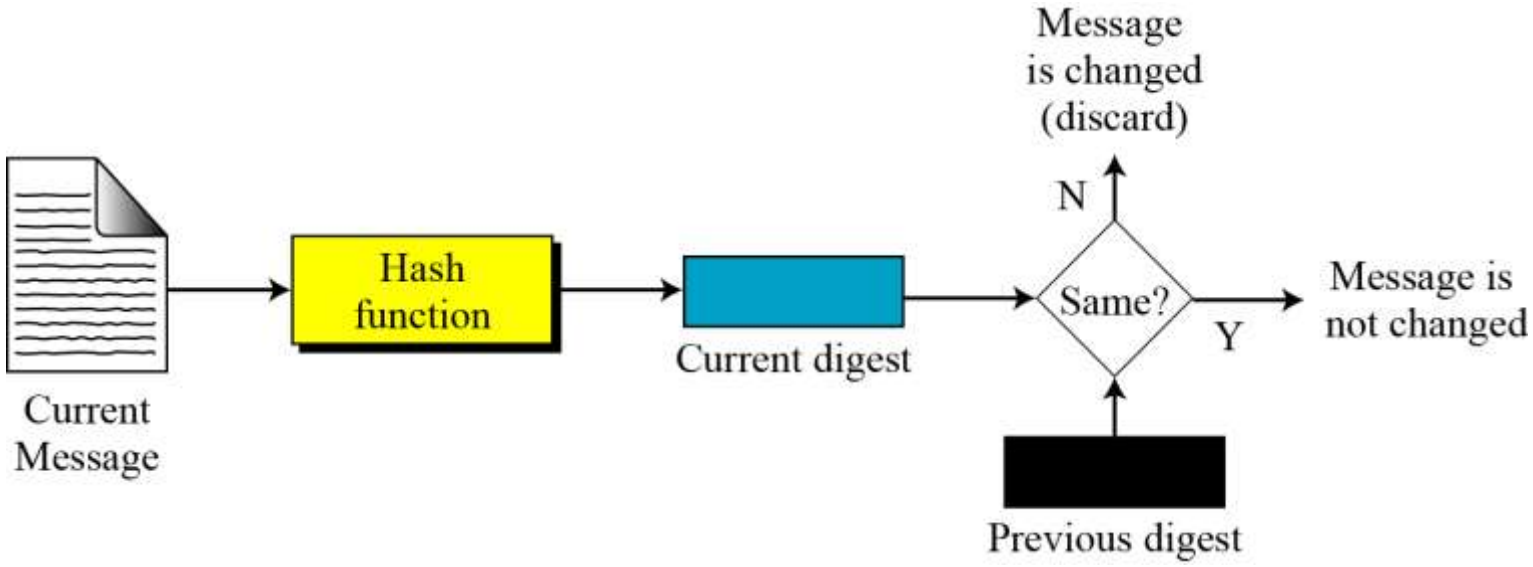


Difference

- The two pairs (document / fingerprint) and (message / message digest) are similar, with some differences.
- The document and fingerprint are physically linked together. The message and message digest can be unlinked separately.
- Most importantly, the message digest needs to be safe from change.

Checking Integrity

Checking integrity



Message Authentication

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- will consider the security requirements
- then three alternative functions used:
  - message encryption
  - message authentication code (MAC)
  - hash function

Message Authentication Functions

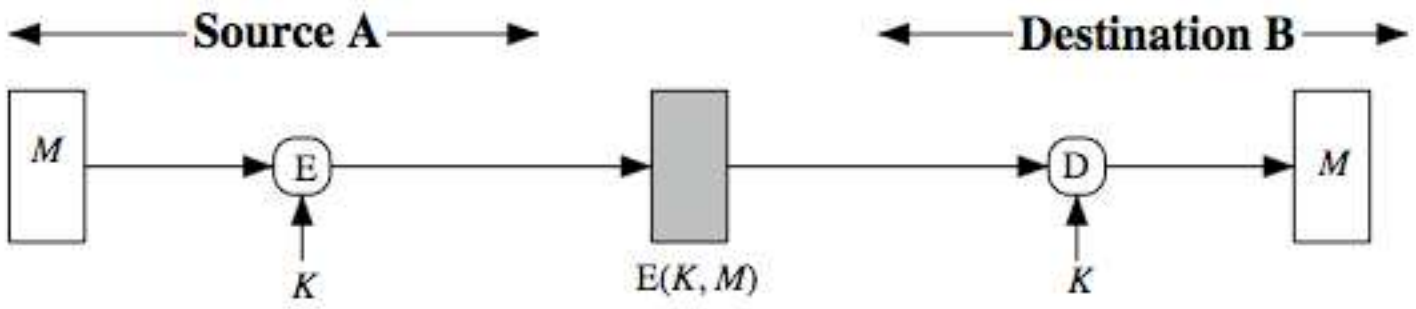
- Message Encryption
- Message Authentication Code (MAC)
- Hash Function

Symmetric Message Encryption

- encryption can also provides authentication
- if symmetric encryption is used then:
  - receiver know sender must have created it

    …since only sender and receiver know key used
  - know content are not altered...

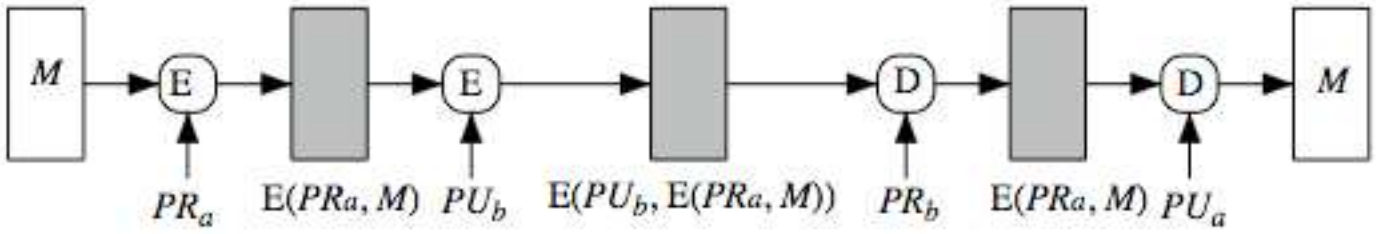
    ... if message has suitable structure, redundancy or a suitable checksum to detect any changes



(a) Symmetric encryption: confidentiality and authentication

Public-Key Message Encryption

- if public-key encryption is used:
  - encryption provides no confidence of sender
    - since anyone potentially knows public-key
  - however if
    - sender **signs** message using their private-key
    - then encrypts with recipients public key
    - have both secrecy and authentication
  - again need to recognize corrupted messages
  - but at cost of two public-key uses on message



(d) Public-key encryption: confidentiality, authentication, and signature

MESSAGE AUTHENTICATION CODE (MAC)

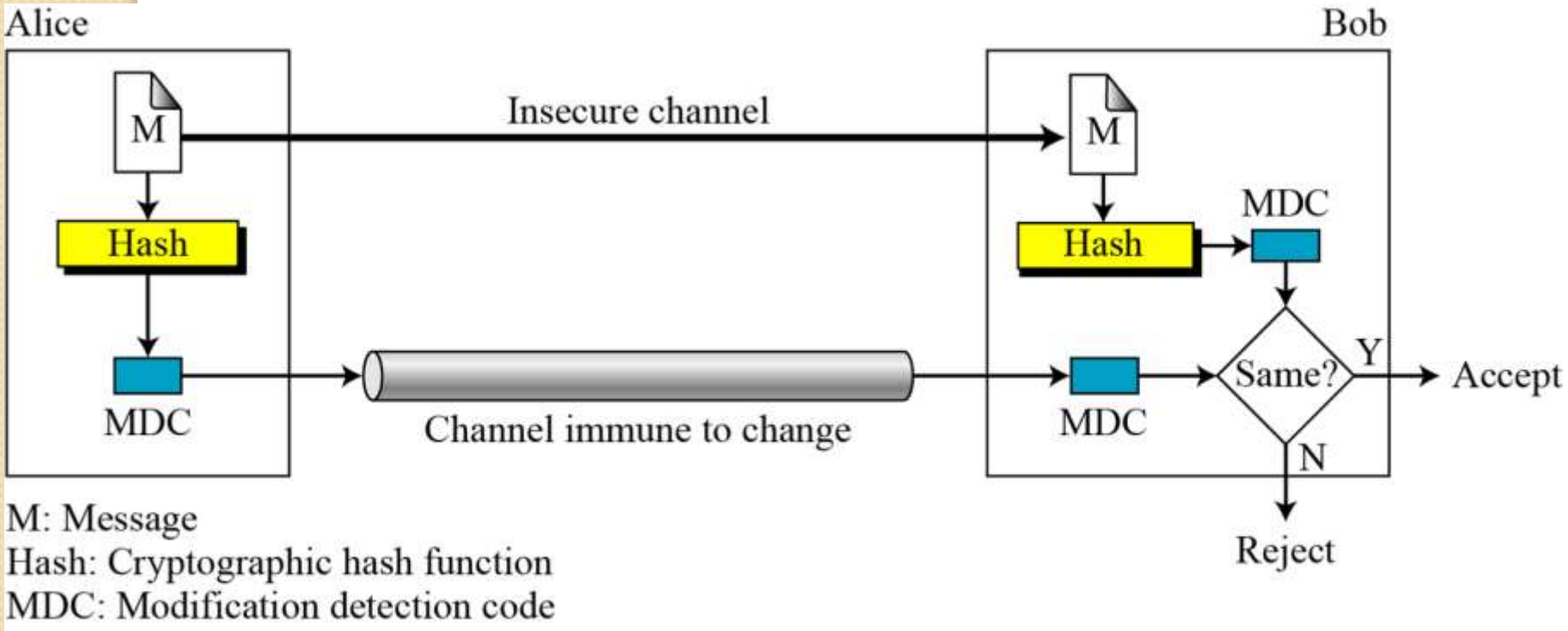
- A message digest does not authenticate the sender of the message.
- To provide message authentication, Alice needs to provide proof that it is Alice sending the message and not an intruder.
- The digest created by a cryptographic hash function is normally called a modification detection code (MDC).
- What we need for message authentication (data origin authentication) is a message authentication code (MAC).

Modification Detection Code (MDC)

- A modification detection code (MDC) is a message digest that can prove the integrity of the message: that message has not been changed.
- If Alice needs to send a message to Bob and be sure that the message will not change during transmission, Alice can create a message digest, MDC, and send both the message and the MDC to Bob.
- Bob can create a new MDC from the message and compare the received MDC and the new MDC. If they are the same, the message has not been changed.

Continued

Modification detection code (MDC)

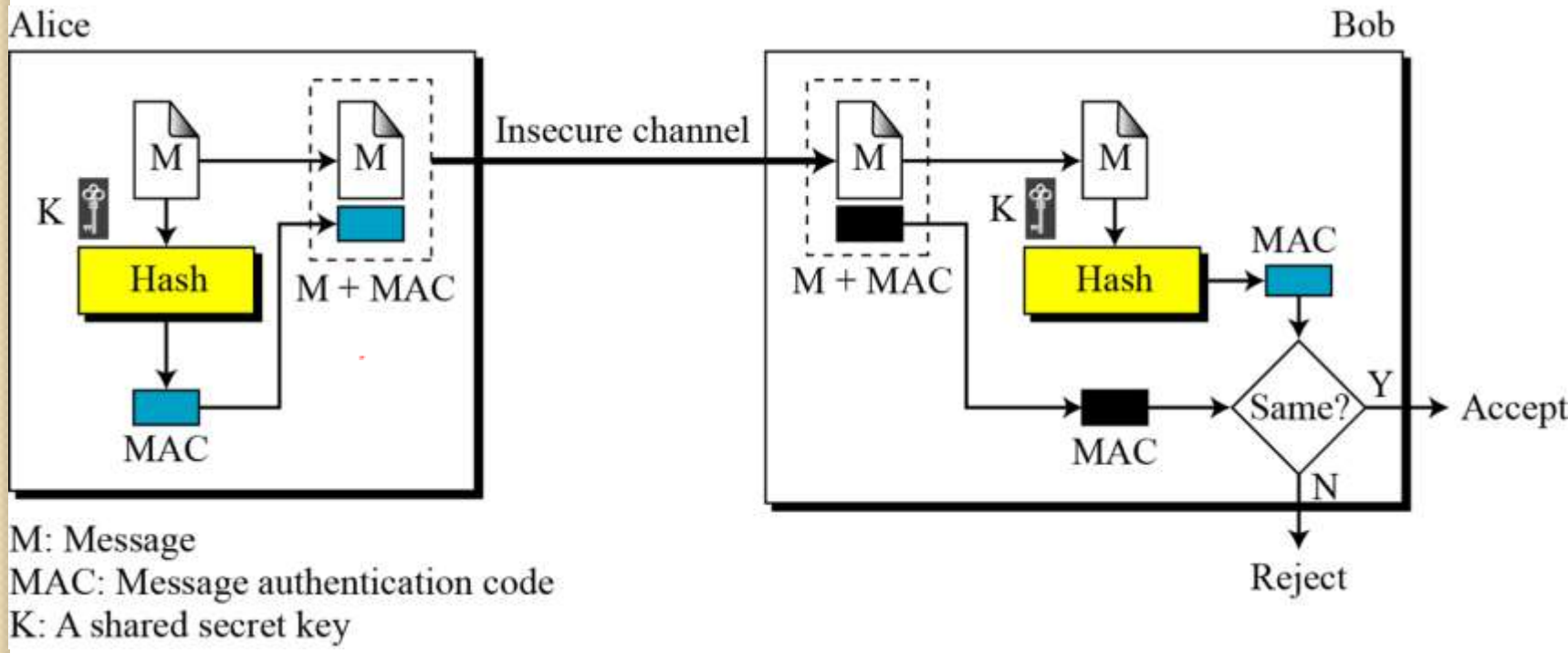


Message Authentication Code (MAC)

- generated by an algorithm that creates a small fixed-sized block
  - depending on both message and secret key
  - like encryption though need not be reversible
- appended to message as a "**signature**"
- receiver performs same computation on message and checks it matches the MAC
- provides assurance that message is unaltered and comes from sender

Message Authentication Code (MAC)

Message authentication code



MAC Properties

- a MAC is a cryptographic checksum

  $\mathbf{MAC = C_K(M)}$

  - condenses a variable-length message M
  - using a secret key K
  - to a fixed-sized authenticator
- is a many-to-one function
  - potentially many messages have same MAC
  - but finding these needs to be very difficult

Message Authentication Codes

- There are three types of MAC:

1. Prefix MAC: secret key is appended to the beginning of the message.
2. Postfix MAC: secret key is appended to the end of message.
3. Combined prefix and postfix MAC: same or two different keys appended to the beginning and end of message

Security of MACs

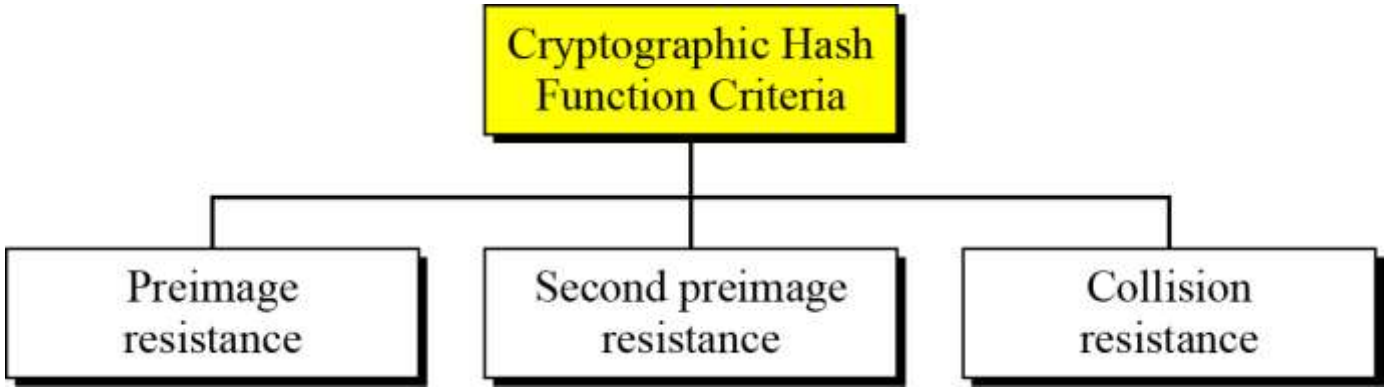
- The security of a MAC depends on the security of the underlying hash algorithm.

Cryptographic Hash Function Criteria

- Hash Functions need to satisfy certain properties, which are essential for the hash functions to be secure and useful in various applications.

Cryptographic Hash Function Criteria

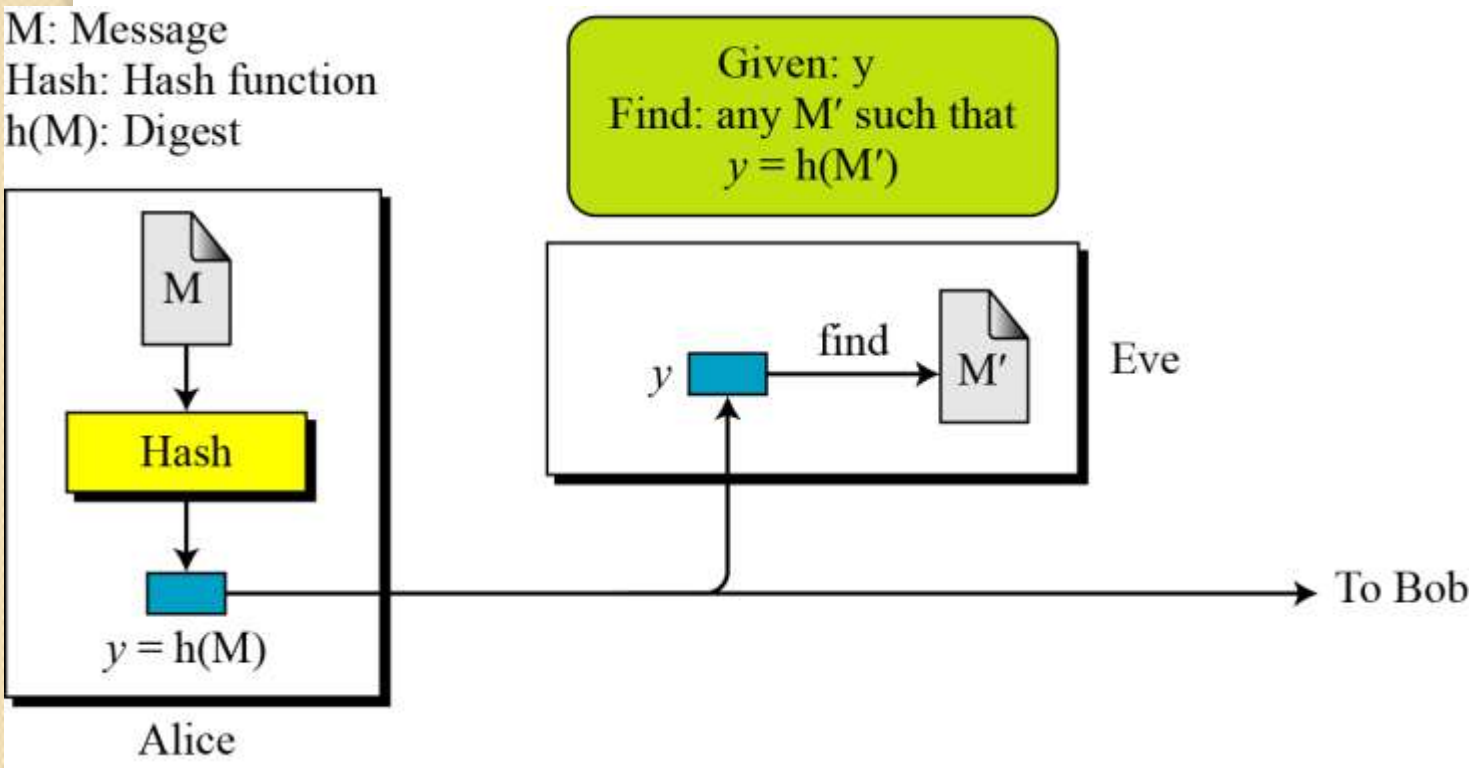
Criteria of a cryptographic hash function



Preimage Resistance

Preimage Attack

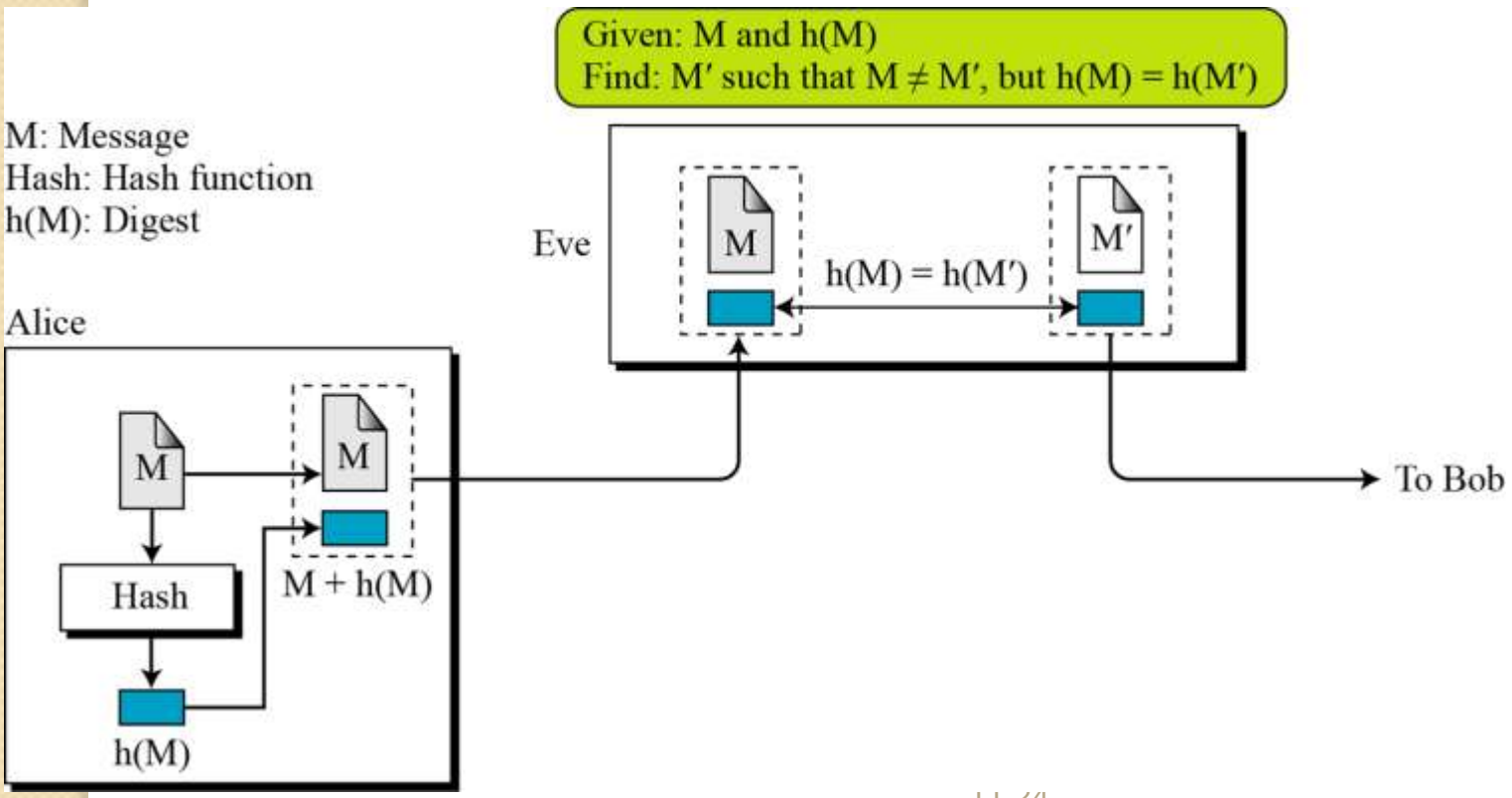
Given: y = h(M) **Find: M′ such that y = h(M′)**



Second Preimage Resistance

Second Preimage Attack

Given: M and h(M) **Find: M′ ≠ M such that h(M) = h(M′)**



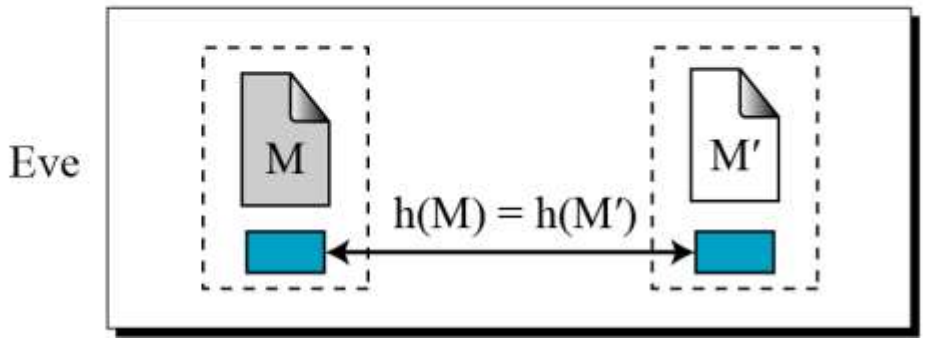
Collision Resistance

Collision Attack

Given: none **Find: M′ ≠ M such that h(M) = h(M′)**

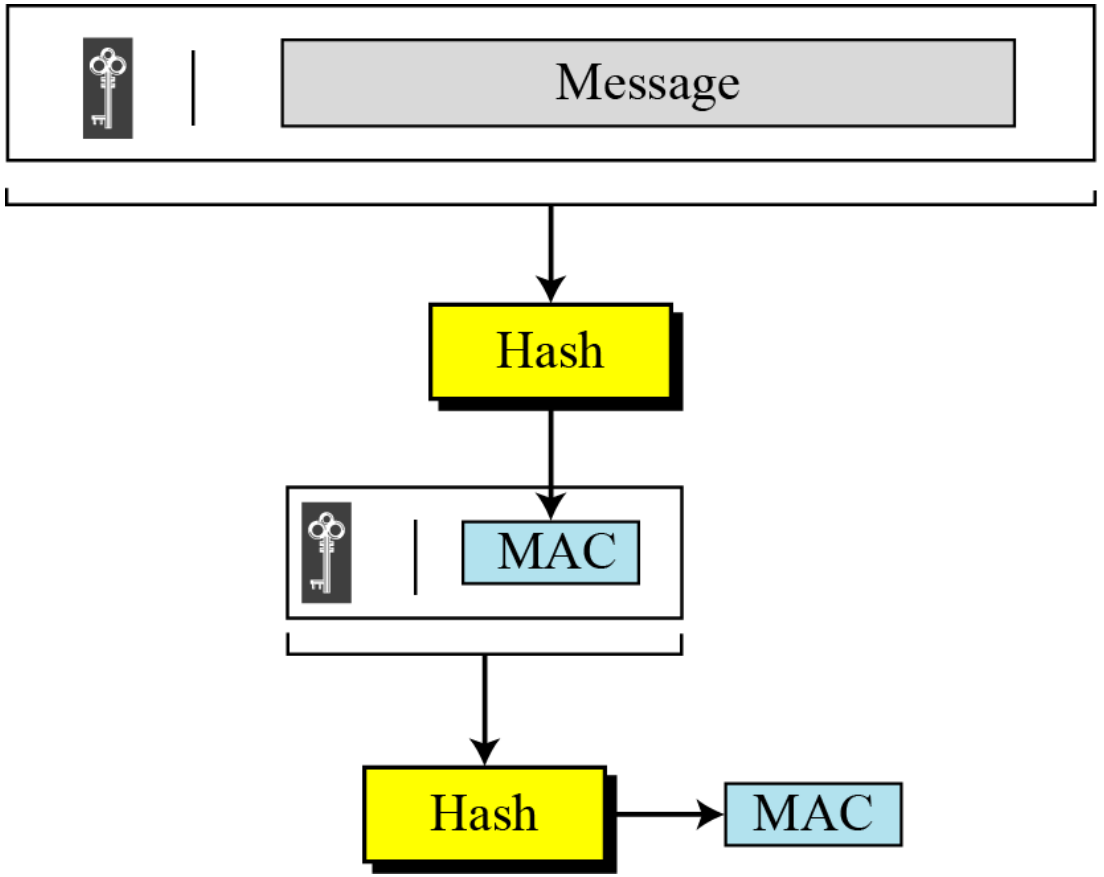
M: Message
Hash: Hash function
h(M): Digest

Find: M and M′ such that M ≠ M′, but h(M) = h(M′)



Nested MAC

Nested MAC

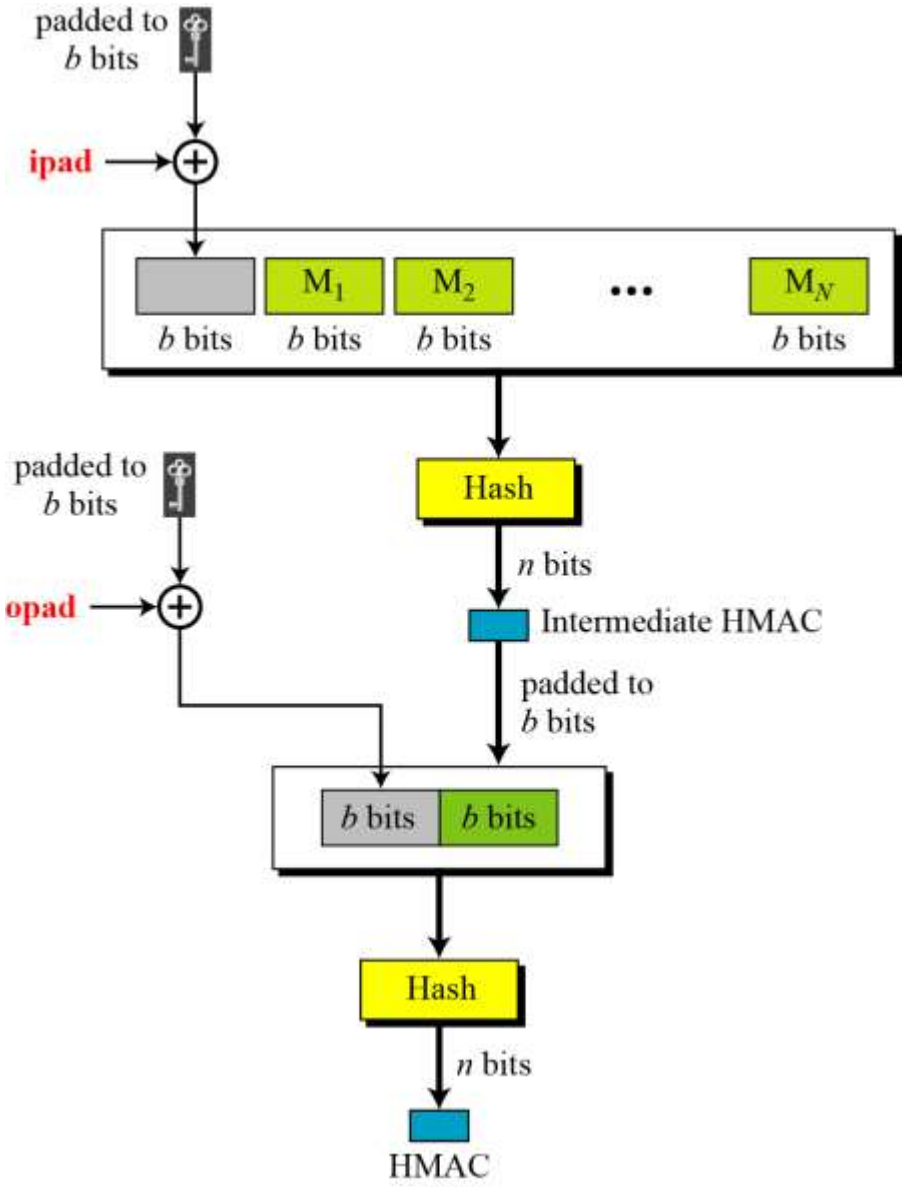


Hashed MAC (HMAC)

- NIST has issued a standard (FIPS 198) for nested MAC that is often referred to as HMAC (Hashed MAC)
- The implementation of HMAC is much more complex than the simplified nested MAC.
- Additional features such as padding

HMAC

Details of HMAC



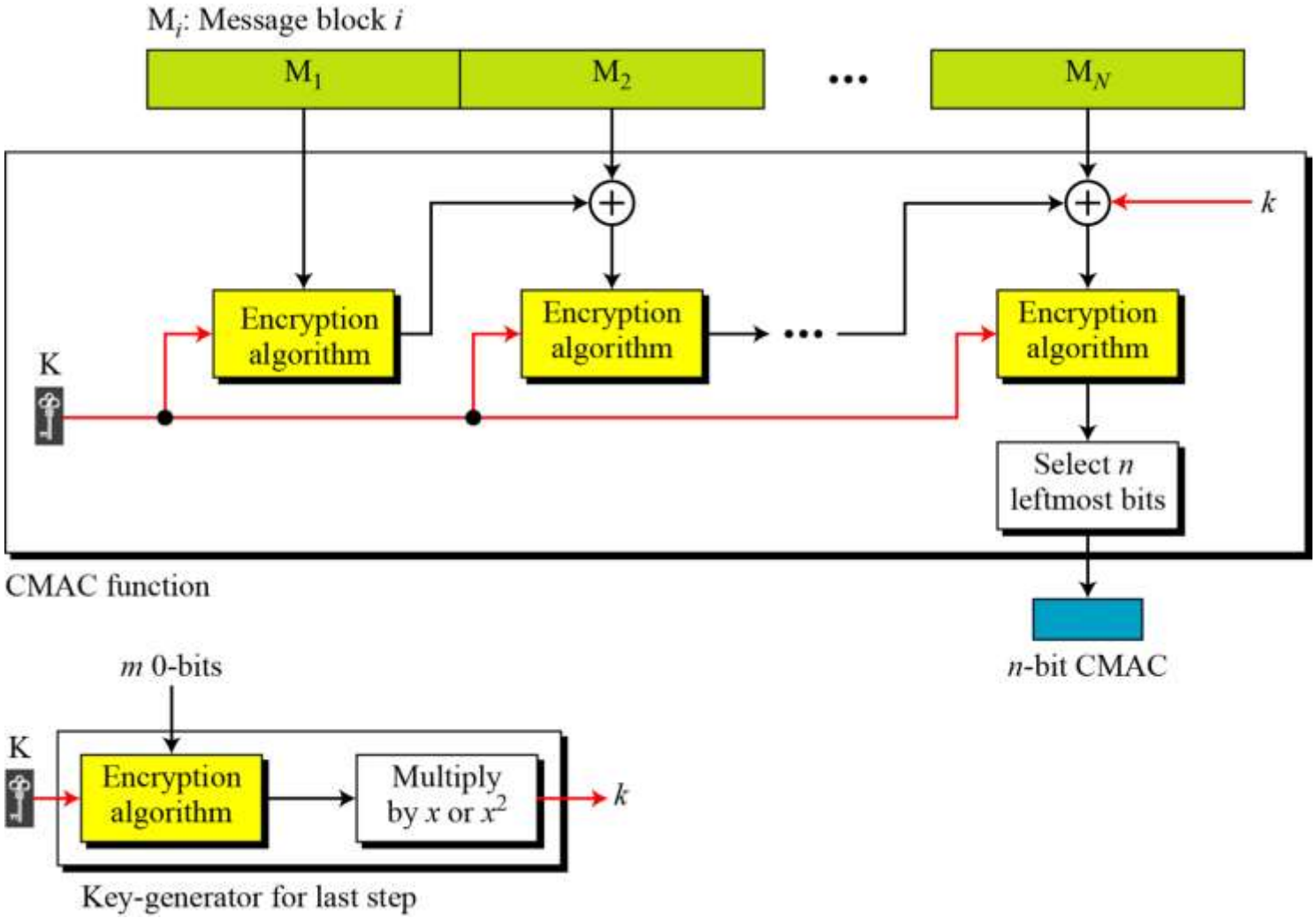
1. The message is divided into N blocks, each of b bits.
2. The secret key is left-padded with 0's to create a b-bit key. Note that it is recommended that the secret key (before padding) be longer than n bits, where n is the size of the HMAC.
3. The result of step 2 is exclusive-ored with a constant called **ipad (input pad)** to create a b-bit block. The value of ipad is the $b/8$ repetition of the sequence 00110110 (36 in hexadecimal).
4. The resulting block is prepended to the N-block message. The result is $N + 1$ blocks.
5. The result of step 4 is hashed to create an n-bit digest. We call the digest the intermediate HMAC.

6. The intermediate n-bit HMAC is left padded with 0s to make a b-bit block.
7. Steps 2 and 3 are repeated by a different constant opad (**output pad**). The value of opad is the $b/8$ repetition of the sequence 01011100 (5C in hexadecimal).
8. The result of step 7 is prepended to the block of step 6.
9. The result of step 8 is hashed with the same hashing algorithm to create the final n-bit HMAC.

Cipher based MAC (CMAC)

- NIST has defined a standard (FIPS 113) called Data Authentication Algorithm (DAA) or CMAC or CBCMAC.
- The method is similar to CBC mode of operation of symmetric block cipher.

CMAC


M_i: Message block i
M_1
M_2
M_N
k
K
Encryption algorithm
Encryption algorithm
Encryption algorithm
Select n leftmost bits
CMAC function
n-bit CMAC
m 0-bits
K
Encryption algorithm
Multiply by x or x^2
k
Key-generator for last step

The message is divided into N blocks, each m bits long. The size of the CMAC is n bits. If the last block is not m bits, it is padded with a 1-bit followed by enough 0-bits to make it m bits. The first block of the message is encrypted with the symmetric key to create an m-bit block of encrypted data. This block is XORed with the next block and the result is encrypted again to create a new m-bit block. The process continues until the last block of the message is encrypted. The n leftmost bit from the last block is the CMAC. In addition to the symmetric key, K, CMAC also uses another key, k, which is applied only at the last step. This key is derived from the encryption algorithm with plaintext of m 0-bits using the cipher key, K. The result is then multiplied by x if no padding is applied and multiplied by x^2 if padding is applied. The multiplication is in GF(2^m) with the irreducible polynomial of degree m selected by the particular protocol used.